CUNY SPS DATA 621 - CTG5 - HW2

Betsy Rosalen, Gabrielle Bartomeo, Jeremy O'Brien, Lidiia Tronina, Rose Koh

March 13, 2019

Contents

Overview

For this homework assignment we were asked to create functions in R to carry out calculations for various classification metrics and to investigate functions in the `caret` and `pROC` packages that provide equivalent results. We were also asked to create graphical output that can be used to evaluate the output of classification models, such as binary logistic regression.

Deliverables should use R functions and the other packages to generate the `classification metrics` for the provided data set.

Question 1 - Download the data

1. *Download the classification output data set (attached in Blackboard to the assignment).*

The data is stored in GitHub in the following location: https://raw.githubusercontent.com/silverrainb/data621proj2/master/classification-output-data.csv. We used the `read.csv()` function to import it directly from GitHub.

Question 2 - Raw confusion matrix

2. *The data set has three key columns we will use:*

- ***class**: the actual class for the observation*
- ***scored.class**: the predicted class for the observation (based on a threshold of 0.5)*
- ***scored.probability**: the predicted probability of success for the observation*

Use the table() function to get the raw confusion matrix for this scored dataset. Make sure you understand the output. In particular, do the rows represent the actual or predicted class? The columns?

Table 1: raw confusion matrix

	0	1
0	119	30
1	5	27

The rows represent the predicted values, while the columns represent the actual values. The row names and column names are 0's and 1's, each representing negative and positive respectively. As such:

Row 0 Column 0	represents values that were predicted negative and were actually negative	True Negatives (TN)
Row 0 Column 1	represents values that were predicted negative and were actually positive	False Negatives (FN)
Row 1 Column 0	represents values that were predicted positive and were actually negative	False Positives (FP)
Row 1 Column 1	represents values that were predicted positive and were actually positive	True Positives (TP)

Question 3 through 8 - Classification metrics function definition

Write a function that takes the data set as a dataframe, with actual and predicted classifications identified, and returns:

3. *Acurracy*
4. *Error rate*
5. *Precision*
6. *Sensitivity(recall)*
7. *Specificity*
8. *F1 score of the predictions*

Verify that you get an accuracy and an error rate that sums to one.

3. Accuracy

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

4. Error Rate

$$\text{Error Rate} = \frac{\text{False Positives} + \text{False Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

5. Precision

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

6. Sensitivity (or Recall)

$$\text{Sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

7. Specificity

$$\text{Specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

8. F1 Score

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

Table 3: Custom Function Output

accuracy	error.rate	precision	sensitivity	specificity	f1
0.8066298	0.1933702	0.7986577	0.9596774	0.4736842	0.8717949

The accuracy (0.8066298) when added to the error rate (0.1933702) is 1.

Question 9 - F1 score bounds

> *9. Before we move on, let's consider a question that was asked: What are the bounds on the F1 score? Show that the F1 score will always be between 0 and 1.(Hint: If $0 < a < 1$ and $0 < b < 1$ then $ab < a$)*

Given the formula for the F1 score...

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

... it can be determined the values of Precision and Sensitivity are of utmost importance.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \text{ , Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Considering the extremes can lead to a more whole answer regarding the bounds on the F1 score. First, if we assume TP has a value of 0...

$$\text{Precision} = \frac{0}{0 + \text{FP}} \text{ , Sensitivity} = \frac{0}{0 + \text{FN}}$$

$$\text{Precision} = \frac{0}{\text{FP}} \text{ , Sensitivity} = \frac{0}{\text{FN}}$$

$$\text{Precision} = 0 \text{ , Sensitivity} = 0$$

Next, if it is assumed TP has a value of 1...

$$\text{Precision} = \frac{1}{1 + \text{FP}} \text{ , Sensitivity} = \frac{1}{1 + \text{FN}}$$

Depending on the values of FP and FN for Precision and Sensitivity respectively, this will always result in a value greater than 0 and less than or equal to 1. This is the case for all non-zero values of TP.

Since it has been proven that Precision and Sensitivity will have values between 0 and 1 inclusively, the same treatment may be applied to the F1 Score.

If Precision and Sensitivity are both 0...

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

$$\text{F1 Score} = \frac{2 \times 0 \times 0}{0 + 0}$$

$$\text{F1 Score} = \frac{0}{0}$$

$$\text{F1 Score} = \text{undefined}$$

If Precision and Sensitivity are both 1...

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

$$\text{F1 Score} = \frac{2 \times 1 \times 1}{1 + 1}$$

$$\text{F1 Score} = \frac{2}{2}$$

$$\text{F1 Score} = 1$$

On each extreme the F1 Score has a value of undefined (or approaching 0) and a value of 1. That means if Precision and Sensitivity should fall between 0 and 1, the F1 Score should also have a value that falls between 0 and 1.

In short, the bounds of the F1 score are 0 and 1.

Question 10 - ROC function definition

10. Write a function that generates an ROC curve from a data set with a true classification column (class in our example) and a probability column (scored.probability in our example). Your function should return a list that includes the plot of the ROC curve and a vector that contains the calculated area under the curve (AUC). Note that I recommend using a sequence of thresholds ranging from 0 to 1 at 0.01 intervals.

A receiver operating characteristic (ROC) curve plots the true positive rate (sensitivity) against the false positive rate (1 - specificity) for all of the possible cutoff values.

$$\text{TPR} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$\text{FPR} = \frac{\text{False Positive}}{\text{False Positive} + \text{True Negative}}$$

Question 11 - Function outputs

*11. Use your **created R functions** and the provided classification output data set to produce all of the classification metrics discussed above.*

Classification metrics custom function output

Table 4: Custom Function Output

accuracy	error.rate	precision	sensitivity	specificity	f1
0.8066298	0.1933702	0.7986577	0.9596774	0.4736842	0.8717949

The output in Table 3 was created using the custom function built to answer questions 3 through 8 above (see the Appendix for function code).

ROC custom function output

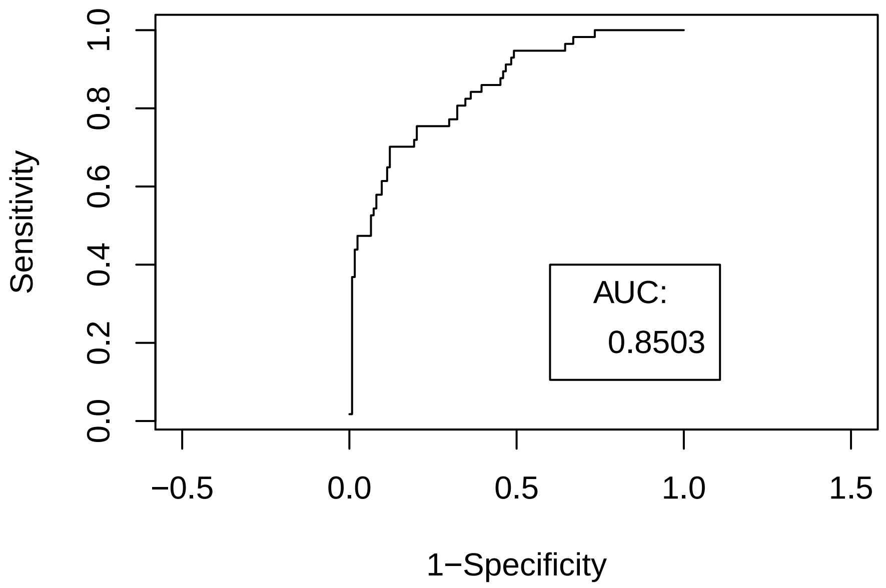

Figure 1: ROC curve using custom function

The ROC curve in Figure 1 was created using the custom function built to answer question 10 above (see the Appendix for function code).

The trajectory of the curve between (0.0) and the 60% threshold is steep, indicating that the sensitivity is increasing at a greater rate than decrease in specificity. However, when the sensitivity is greater than 80%, there is a more significant decrease in specificity than the gain in sensitivity.

The area under the ROC curve (AUC) is a measure, of how well a parameter can be distinguished between two diagnostic groups. Computing the area under the curve is one way to summarize it in a single value. Our model has a good calculated AUC: 0.8503. It's near 1, which means it has good measure of class separation.

Question 12 - `caret` package

> *12. Investigate the **caret** package. In particular, consider the functions confusionMatrix, sensitivity, and specificity. Apply the functions to the data set. How do the results compare with your own functions?*

The caret ("Classification and Regression Training") package is a toolset for model training. Its function `confusionMatrix` provides a shortcut to cross-tabulate observed and predicated classes with their statistics, just like the custom function created in the preceding questions.

`confusionMatrix` takes as input the actual and predicted class for each observation - both as factors with the same set of levels, or as a table (the approach we used). After computing the prevalence of true and false positives and negatives, the function outputs a list of statistics (including several the custom function was not asked to examine, for which formulae are included below):

- Accuracy
- 95% confidence interval and p-value
- No information rate (rate at which correct if just selected majority class) and its p-value
- Kappa coefficient (a measure agreement between actual and predicted classes comparing to probability of chance agreement)

- McNemar's test p-value
- Sensitivity
- Specificity
- PPV (Positive Predicted Value)

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- NPV (Negative Predicted Value)

$$\frac{\text{True Negatives}}{\text{True Negatives} + \text{False Negatives}}$$

- Prevalence

$$\frac{\text{True Positives} + \text{False Negatives}}{\text{True Positives} + \text{False Positives} + \text{True Negatives} + \text{False Negatives}}$$

- DetectionRate

$$\frac{\text{True Positives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

- DetectionPrevalence

$$\frac{\text{True Positives} + \text{False Positives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

- Balanced Accuracy

$$\frac{\text{Sensitivity} + \text{Specificity}}{2}$$

The caret packages `ConfusionMatrix` yielded identical results to the statistics computed by the custom function to seven decimal places.

Table 5: Custom Function Output

accuracy	error.rate	precision	sensitivity	specificity	f1
0.8066298	0.1933702	0.7986577	0.9596774	0.4736842	0.8717949

Table 6: caret Package confusionMatrix Output

accuracy	error.rate	precision	sensitivity	specificity	f1
0.8066298	0.1933702	0.7986577	0.9596774	0.4736842	0.8717949

Question 13 - `pROC` package

*13. Investigate the **pROC** package. Use it to generate an ROC curve for the data set. How do the results compare with your own functions?*

The ROC curve in the figure below was created using the `pRoc` Package.

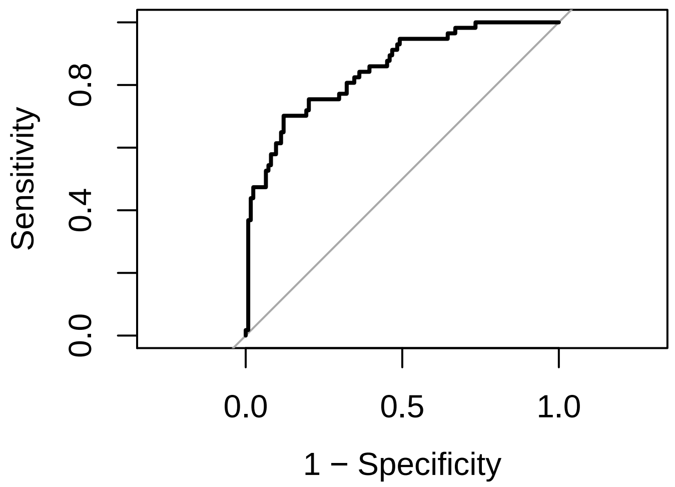

Figure 2: ROC curve from pROC package

Comparing this curve to the one we created using the custom function, we can see that the two curves look identical.

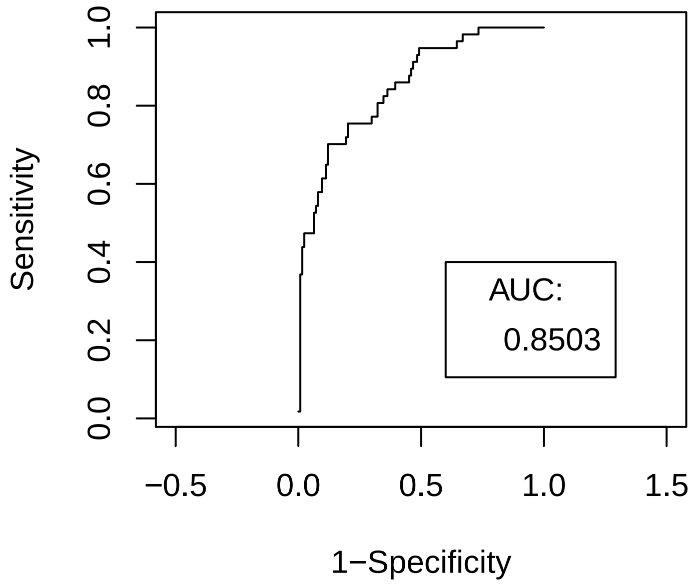

Figure 3: ROC curve using custom function

We also got matching results for the area under the curve (AUC): 0.8503.

Using the `pROC` package we can also compute the confidence interval (CI) of a ROC curve. By default, it's calculated at 95% confidence. The computed confidence interval for our curve is 0.7905 - 0.9101.

Appendix

```
# Q1
raw.data <- read.csv('https://raw.githubusercontent.com/silverrainb/data621proj2/
master/classification-output-data.csv', stringsAsFactors = F, header = T)
data <- as.data.table(raw.data)

# Q2

q2.tbl <- table(data$scored.class,
             data$class)

# Q3 - Q8

ClassCalc <- function(tbl) {
  # with tbl, get TP, FP, FN, TN
  tp <- tbl[1,1];
  fp <- tbl[1,2];
  fn <- tbl[2,1];
  tn <- tbl[2,2];

  # calculate summary stats
  accuracy <- (tp+tn) / (tp+fp+tn+fn)
  error.rate <- (fp+fn) / (tp+fp+tn+fn)
  precision <- tp / (tp+fp)
  sensitivity <- tp / (tp+fn)
  specificity <- tn / (tn+fp)
  f1 <- (2 * precision * sensitivity) / (precision + sensitivity)

  # create dataframe
  df <- data.frame(accuracy = accuracy,
                   error.rate = error.rate,
                   precision = precision,
                   sensitivity = sensitivity,
                   specificity = specificity,
                   f1 = f1)
  return(df)
}

q3.q8.output <- ClassCalc(q2.tbl)

# Q 10

ROC_func <- function(labels, scores){
    labels <- labels[order(scores, decreasing=TRUE)]
    result <- data.frame(TPR=cumsum(labels)/sum(labels),
                         FPR=cumsum(!labels)/sum(!labels), labels)
    dFPR <- c(diff(result$FPR), 0)
    dTPR <- c(diff(result$TPR), 0)
    AUC <- round(sum(result$TPR * dFPR) + sum(dTPR * dFPR)/2,4)
    plot(result$FPR,result$TPR,type="l",ylab="Sensitivity",xlab="1-Specificity",
                         xlim = c(-.5,1.5))
    legend(.6,.4,AUC,title = "AUC: ")
}
```

```
q10roc <- ROC_func(data$class,data$scored.probability)


#Q12

# Create dataframe for Q12, subsetting key columns: class, scored.class,
# scored.probability

q12.data <- data %>%
    dplyr::select(class, scored.class, scored.probability)

# Create confusion matrix using caret package
q12.confmtrx <- caret::confusionMatrix(data = factor(q12.data$scored.class),
                                        # predicted classes of observations in factor form
                                       reference = factor(q12.data$class))
                                        # actual classes of observations in factor form

# reference doc:
# https://stackoverflow.com/questions/34842837/saving-output-of-confusionmatrix-as-a-csv-table

# Extract classification results of interest from caret::confusionMatrix object
# for comparison to custom function results

q12.output <- cbind(t(q12.confmtrx$overall),t(q12.confmtrx$byClass)) %>%
    data.frame() %>%
    mutate(error.rate = (1 - Accuracy)) %>%
    select(Accuracy, error.rate, Precision, Sensitivity, Specificity, F1) %>%
    janitor::clean_names() %>%
    rename(error.rate = error_rate)

# Table of custom function results
knitr::kable(q3.q8.output)

# Table of caret::confusionMatrix results
knitr::kable(q12.output)

# Compare results - evaluates as equal
all.equal(q12.output, q3.q8.output)


# Q 13

library(pROC)
rocCurve <- roc(data$class , data$scored.probability)
q13roc <- plot(rocCurve , legacy.axes = TRUE,   main = "pROC Package")

auc13 <- auc(rocCurve)
ci13 <- ci(rocCurve)
```